

# STOCHASTIC SHORTEST PATHS AND RISK MEASURES

AXEL PARMENTIER AND FRÉDÉRIC MEUNIER

**ABSTRACT.** We consider three shortest path problems in directed graphs with random arc lengths.

For the first and the second problems, a risk measure is involved. While the first problem consists in finding a path minimizing this risk measure, the second one consists in finding a path minimizing a deterministic cost, while satisfying a constraint on the risk measure. We propose algorithms solving these problems for a wide range of risk measures, which includes among several others the *CVaR* and the probability of being late. Their performances are evaluated through experiments.

One of the key elements in these algorithms is the use of stochastic lower bounds that allow to discard partial solutions. Good stochastic lower bounds are provided by the so-called *STOCHASTIC ON TIME ARRIVAL PROBLEM*. This latter problem is the third one studied in this paper and we propose a new and very efficient algorithm solving it.

Complementary discussions on the complexity of the problems are also provided.

## 1. INTRODUCTION

**1.1. Context.** Finding shortest paths in networks has numerous applications. There are direct applications, e.g. organizing the routes of a fleet in logistics or finding the best trajectory for an airplane, but there are also less obvious applications, such as the ones arising as subproblems in column generation approaches.

As for other optimization problems, there is currently a need for taking into account uncertainty in shortest path problems, both from theoretical and practical point of views. Taking into account the uncertainties allows a better optimization. For instance, congestion in road networks has a strong influence on travel times, and modeling delay is crucial when on time arrival is required. Several works have already been carried out on this topic – see our literature review in Section 2 – but as far as we know, none deal with the problems expressed in the general form we address in the present paper.

**1.2. Problems.** We consider various shortest path problems with random arc lengths. They all have in common the following elements: a directed graph  $D = (V, A)$  without loops, two vertices  $o, d \in V$  that are respectively the origin and the destination, and independent random variables  $(X_a)_{a \in A}$  that model random travel times. The objective is always to find the best  $o$ - $d$  path. ‘Best’ can take various meanings and we adopt a versatile approach.

The most natural case consists probably in taking no additional constraints into account. In order to be as general as possible, we study the problem of minimizing a given *risk measure* of the sum of the  $X_a$ ’s along the path. A risk measure is a mapping from a set of random variables to the set of real numbers. The mean, the variance, or the probability of being above some threshold are examples of risk measures. Another risk measure is the conditional value at risk and is well known in finance but maybe less in operations research. It has many interesting features. Details are given later in the paper.

Given a risk measure  $\rho$ , we define the following problem.

---

*Key words and phrases.* On time arrival, random travel time, risk measure, resource constraint, shortest path, usual stochastic order.

#### STOCHASTIC $\rho$ -SHORTEST PATH PROBLEM

**Instance.** A directed graph  $D = (V, A)$ , two vertices  $o, d \in V$ , independent random variables  $(X_a)_{a \in A}$ .

**Solution.** An elementary  $o$ - $d$  path  $P$ .

**Measure.** The quantity  $\rho(\sum_{a \in P} X_a)$ .

When the  $X_a$  are deterministic and  $\rho$  is linear, we obtain the usual (deterministic) shortest path problem.

Another way to take into account stochasticity when modeling an optimization problem consists in introducing probabilistic constraints. Still assuming given a risk measure  $\rho$ , we can define the following problem.

#### STOCHASTIC $\rho$ -RESOURCE CONSTRAINED SHORTEST PATH PROBLEM

**Instance.** A directed graph  $D = (V, A)$ , two vertices  $o, d \in V$ , independent random variables  $(X_a)_{a \in A}$ , a nonnegative number  $\rho_0$ , and nonnegative numbers  $(c_a)_{a \in A}$ .

**Solution.** An elementary  $o$ - $d$  path  $P$  such that  $\rho(\sum_{a \in P} X_a) \leq \rho_0$ .

**Measure.** The quantity  $\sum_{a \in P} c_a$ .

When the  $X_a$  are deterministic and  $\rho$  is linear, we obtain the usual (deterministic) resource constrained shortest path problem.

**1.3. Assumption on the travel times  $X_a$ .** All random variables  $X_a$ , which model the time needed to traverse an arc  $a$ , are assumed to take their values in  $\mathbb{Z}_+$  and to have finite supports. In a computational perspective, it is a natural assumption. When complexity features are discussed, the variable  $X_a$  is described by its distribution, which is assumed to be encoded as its support and the probability of each value in the support.

**1.4. Contributions.** We propose an algorithm for each of the aforementioned problems when the arc lengths  $X_a$  satisfy the assumption above and when the risk measure  $\rho$  penalizes delay. The mean, the probability of being above some threshold, and the conditional value at risk are examples of such risk measures, but not the variance. “Penalizing delay” will be mathematically specified with the help of the so-called usual stochastic order, which is an order defined on the set of random variables – see Section 3. However, our approach remains completely natural. The efficiency of our algorithms, which turns out to be high, is proved by an extensive experimental study.

One of the main ideas used in these algorithms consists in attaching to each vertex  $v$  of the graph a random variable  $Z_v$  that is a lower bound of the length of a  $v$ - $d$  path. “Lower bound” is here to be understood in the sense of the usual stochastic order. As usual with lower bounds in shortest paths algorithms, these stochastic lower bounds allow to discard a priori some partial paths and to reduce the space of possible solutions.

It turns out that good lower bounds are solutions of another problem, namely the STOCHASTIC ON TIME ARRIVAL PROBLEM. There are already efficient algorithms solving this problem, but we propose a new one, which seems to be very efficient.

The complexity of the problems dealt with in the paper are also discussed.

**1.5. Plan.** We start with a literature review (Section 2). We describe in Section 3 the usual stochastic order and the various risk measures covered by our work. The stochastic lower bounds used in our algorithms and their links with the STOCHASTIC ON TIME ARRIVAL PROBLEM are described in Section 4, as well as the new algorithm we propose for this latter problem. The algorithms for the STOCHASTIC  $\rho$ -SHORTEST PATH PROBLEM and the STOCHASTIC  $\rho$ -RESOURCE CONSTRAINED SHORTEST PATH PROBLEM are described in Sections 5 and 6 respectively, as well as their complexity analysis. The paper ends with an extensive experimental study (Section 7).

## 2. LITERATURE REVIEW

**2.1. Deterministic shortest path.** The shortest path problem with deterministic travel times is one of the most studied problems in Operations Research. Efficient polynomial algorithms are known since the end of the fifties. Recently, research has focused on improving algorithm efficiency thanks to pre-processing, and state of the art techniques compute shortest paths on continental road networks in fractions of a microsecond. An exhaustive survey on shortest path algorithms for route planning can be found in (Bast *et al.* , 2014).

**2.2. Deterministic constrained and multicriteria shortest path.** Deterministic shortest paths with resource constraints have many direct applications in logistics and transportation. Moreover, many operations research problems require to find such shortest paths as a subroutine (e.g. vehicle routing problems). Resource constrained shortest path have therefore been extensively studied, and a detailed literature review is out of the scope of this paper. A review of state of the art techniques can be found in (Irnich & Desaulniers, 2005).

Several problems close to the resource constrained shortest path problem belong to multicriteria optimization (Ehrgott, 2005). Many algorithms have been developed for the bicriteria (Raith & Ehrgott, 2009) and multicriteria shortest path problem (Tarapata, 2007). State of the art solution methods for resource constrained or multicriteria shortest path problems often rely on labeling algorithms with pre-processing (Boland *et al.* , 2006; Dumitrescu & Boland, 2003).

**2.3. Stochastic shortest path.** Stochastic shortest path problems have been extensively studied since the seminal work of Frank (1969). Models differ by the probability distributions used to model delay on arcs, and by the risk measure optimized.

A first line of papers considers the probability of on time arrival: a path maximizing the probability of on time arrival, or analogously, a path with minimum quantile of given order is searched. Approaches have been developed for both continuous (Chen & Ji, 2005; Frank, 1969; Nikolova, 2010; Nikolova *et al.* , 2006) and discrete distributions (Mirchandani, 1976). An efficient labeling algorithm has recently been described when arc distributions are normal (Chen *et al.* , 2013). In a recent preprint, Niknami *et al.* (2014) proposes independently an idea that bears similarities with ours by using the solution of the STOCHASTIC ON TIME ARRIVAL PROBLEM to obtain lower bounds for the STOCHASTIC  $\rho$ -SHORTEST PATH PROBLEM with  $\rho(\cdot) = \mathbb{P}(\cdot \geq \tau)$ . Note that our approach covers actually more general situations, even in this special case.

A second line of papers defines a shortest path as a path minimizing the expectation of a cost function (Loui, 1983). Dynamic programming can be used when cost functions are affine or exponential (Eiger *et al.* , 1985). Murthy & Sarkar (1996, 1998) present an efficient labeling algorithm when arc distributions are normal and cost functions are piecewise-linear and concave. Instead of considering the expectation of a cost functions, Nikolova (2010); Nikolova *et al.* (2006); Sivakumar & Batta (1994) search the path that minimizes a positive linear combination of mean and variance.

Finally, Miller-Hooks & Mahmassani (2003) suggest to use stochastic dominance to compare paths. Algorithms to generate all non-dominated paths are proposed in Miller-Hooks & Mahmassani (2003, 1998); Miller-Hooks (1997); Nie & Wu (2009); Nie *et al.* (2012).

**2.4. Stochastic on time arrival.** The STOCHASTIC ON TIME ARRIVAL PROBLEM searches to maximize the probability of arrival before a given thresholds on adaptive paths (Fu, 2001; Fu & Rilett, 1998; Hall, 1986). An adaptive path is an application which, given a vertex reached and the time it took to get to this vertex indicates the next arc to choose to maximize the probability of on time arrival at destination. Fan & Nie (2006) provide an algorithm for continuous distributions, and Nie & Fan (2006) provide a pseudo-polynomial algorithm for discrete distributions. Samaranayake *et al.* (2012) develop a faster algorithm for discrete distributions, and Sabran *et al.* (2014) provide pre-processing techniques to improve algorithm speed.

**2.5. Shortest path under probability constraint.** Finally, the problem of finding a minimum cost path for deterministic arc costs under stochastic resource constraints have been introduced in Kosuch & Lissner (2010), which proposes a solution algorithm based on linear programming is derived.

### 3. USUAL STOCHASTIC ORDER AND RISK MEASURES

**3.1. Usual stochastic order.** For any random variable  $X$ , let  $F_X(t) = \mathbb{P}(X \leq t)$  be its cumulative distribution. The *usual stochastic order*  $\leq_{st}$  is a partial order defined on the set of cumulative distributions as follows:  $F_X \leq_{st} F_Y$  if  $F_X(t) \geq F_Y(t)$  for all  $t$ . Endowed with the usual stochastic order, the set of cumulative distributions turns out to be a *lattice*. It means that each collection of cumulative distributions has a unique least upper bound (*join*) and a unique greatest lower bound (*meet*).

We use the same notation  $\leq_{st}$  to extend the usual stochastic order to random variables. Two random variables  $X$  and  $Y$  are such that  $X \leq_{st} Y$  if  $F_X \leq_{st} F_Y$ . Note that in the context of random variables, it is a *quasiorder*: it is reflexive and transitive like a partial order, but not antisymmetric. The usual stochastic order is a classical notion in probability theory and enjoys several properties. Missing details and a more complete introduction may be found in Müller & Stoyan (2002).

One property is particularly useful in our work and is used in the paper without further mention.

*Let  $X_1, X_2, Y_1$ , and  $Y_2$  be random variables such that  $X_1$  and  $Y_1$  are independent, and  $X_2$  and  $Y_2$  are independent. If  $X_1 \leq_{st} X_2$  and  $Y_1 \leq_{st} Y_2$ , then  $X_1 + Y_1 \leq_{st} X_2 + Y_2$ .*

By a slight abuse, we define the meet (resp. the join) of a collection  $(X_i)$  of random variables as a random variable with cumulative distribution equal to the meet (resp. the join) of the  $F_{X_i}$ 's. We use the classical notation  $\wedge$  for the meet and  $\vee$  for the join. We have in particular for any collection  $(X_i)$  of random variables the following equality

$$F_{\bigwedge_i X_i}(t) = \max_i(F_{X_i}(t)) \quad \text{for all } t \in \mathbb{Z}_+.$$

The meet of the empty collection is as usual equal to  $+\infty$  and its join is equal to  $-\infty$ .

Since the supports of the random variables are assumed to be integer valued and of finite cardinality, their meet (resp. the join) can easily be computed: for each  $t$  in the union of their supports, we simply keep the largest (resp. smallest) value among those taken by the cumulative distributions at  $t$ . In the paper, we only use the meet operation.

**3.2. Risk measures.** A risk measure  $\rho$  is said to be *consistent with the usual stochastic order* if  $X \leq_{st} Y$  implies  $\rho(X) \leq_{st} \rho(Y)$ . The algorithms for the STOCHASTIC  $\rho$ -SHORTEST PATH PROBLEM and the STOCHASTIC  $\rho$ -RESOURCE CONSTRAINED SHORTEST PATH PROBLEM introduced in this paper only require the consistency of the risk measure with the usual stochastic order. It is not a strong assumption since being consistent with the risk measure simply means penalizing delay. Note however that the variance does not satisfy this condition.

For examples, the following risk measures are covered by our work.

- $\rho(\cdot) = \mathbb{E}[f(\cdot)]$  for any increasing function  $f$ .
- $\rho(\cdot) = \mathbb{P}(\cdot \geq \tau)$  for any  $\tau \in \mathbb{Z}_+$ .
- $\rho(\cdot) = VaR_\beta(\cdot)$  for any  $\beta \in [0, 1]$ .
- $\rho(\cdot) = CVaR_\beta(\cdot)$  for any  $\beta \in [0, 1]$ .

The first risk measure among the ones listed can be used for instance when the cost of being late is modeled. We can cite as an illustration the case of goods delivery, an increasing step function can model the successive penalties of a late delivery.

The second one is typically used to measure the probability of going beyond some deadline. It corresponds to the risk measures of a traveler going to the airport: the only objective is to arrive before the time  $\tau$  when boarding gate is closed.

The third and fourth risk measures are widely used in finance.

$VaR_\beta(\cdot)$  is the *value at risk (with confidence level  $\beta$ )* and is defined by

$$VaR_\beta(X) = \min\{t \in \mathbb{Z}_+ : \mathbb{P}(X \leq t) \geq \beta\}.$$

Given a confidence level  $\beta$ , a random variable  $X$  is smaller than its value at risk  $VaR_\beta(X)$  with probability at least  $\beta$ , and  $VaR_\beta(X)$  is the threshold value for which this inequality holds. For instance, if a bus company advertises the policy “our buses are on time for  $\beta\%$  of the trips”, then the shortest path with respect to  $VaR_\beta$  gives the smallest travel time for which this policy holds.

$CVaR_\beta(\cdot)$  is the *conditional value at risk (with confidence level  $\beta$ )* and is defined by

$$CVaR_\beta(X) = \mathbb{E}[X | X \geq VaR_\beta(X)].$$

Given a confidence level  $\beta$ , the conditional value at risk of a random variable  $X$  can be described in a somehow informal way as its average value in the  $1 - \beta$  worst cases. In the context of shortest paths with random travel times, a  $\beta$  equal to 0 corresponds to a shortest path problem where expectation is minimized. A  $\beta$  equal to 1 corresponds to a robust approach: the optimal solution is the solution with minimal worst case behavior. As a consequence,  $CVaR_\beta$  enables to obtain a tradeoff between expectation and robustness. This cost function suits well most traveler behavior: a small path with small  $CVaR$  gives a small average traveling time with a low risk of very large travel time.

#### 4. STOCHASTIC LOWER BOUNDS AND ON TIME ARRIVAL

**4.1. Stochastic lower bounds.** Efficient shortest path algorithms are often based on lower bounds that allow to discard partial paths and to reduce the space of solutions that is explored. We do not depart from it and introduce stochastic lower bounds, which, instead of being single values, are random variables.

We consider the following equation, where  $\delta^+(v)$  denotes the set of arcs of the form  $(v, u)$ .

$$(1) \quad \begin{cases} Z_d = 0, \\ Z_v \stackrel{(d)}{=} \bigwedge_{(v,u) \in \delta^+(v)} (X_{(v,u)} + Z_u) \quad \text{for all } v \in V \setminus \{d\}. \end{cases}$$

Proposition 3 below shows that this equation has always a solution which can be calculated in polynomial time. Before stating and proving this proposition, we explain the interest of this equation for stochastic shortest path problems.

**Proposition 1.** *Let  $(Z_v)_{v \in V}$  be a solution of Equation (1). For any vertex  $v \in V$  and any  $v$ - $d$  path  $P$ , the inequality  $Z_v \leq_{st} \sum_{a \in P} X_a$  holds.*

*Proof.* An induction on the number of arcs in  $P$  allows to conclude.  $\square$

**4.2. On Time Arrival.** The STOCHASTIC ON TIME ARRIVAL PROBLEM aims at finding the path in  $D$  that maximizes the probability of reaching  $d$  before some time limit  $\tau \in \mathbb{Z}_+$ . If the solution has to be computed a priori, we are exactly in the case of the STOCHASTIC  $\rho$ -SHORTEST PATH PROBLEM, with  $\rho(X) = \mathbb{P}(X \geq \tau)$ , which is a risk measure covered by the present work. In the STOCHASTIC ON TIME ARRIVAL PROBLEM, the path can be built during the trip. Concretely, an optimal policy has to be computed. A policy  $\pi$  for this problem associates to each pair  $(v, t) \in V \times \mathbb{Z}_+$  an arc  $a \in \delta^+(v)$ . Such a policy  $\pi$  is optimal if it maximizes the probability of reaching  $d$  before  $\tau$ . If there is in  $D$  a circuit  $C$  with  $\mathbb{P}(X_a = 0) = 1$  for all  $a \in C$ , the problem may not be well-defined. We suppose therefore that there are no such circuits. Under this assumption, and using elementary properties of Markov chains, we get that solving STOCHASTIC ON TIME ARRIVAL PROBLEM is

equivalent to finding a solution of the following equation for all  $t \in \mathbb{Z}_+$  (see also Fan *et al.* (2005) for a first formulation of this kind in a continuous setting):

$$(2) \quad \begin{cases} F_d(t) = 1 \\ F_v(t) = \max_{(v,u) \in \delta^+(v)} \sum_{k=0}^t \mathbb{P}(X_{(v,u)} = k) F_u(t-k) \quad \text{for all } v \in V \setminus \{d\}. \end{cases}$$

The quantity  $F_v(t)$  is then the probability to reach  $d$  from  $v$  in less than time  $t$  under an optimal policy. The decision to be taken at time 0 on vertex  $o$  is determined by the  $(o, u) \in A$  for which the maximum is attained when  $v = o$  and  $t = \tau$  in Equation (2).

Given a realization of the random variables  $X_a$ , the path obtained following the policy of Equation (2) is not necessary elementary. Samaranayake *et al.* (2012) provides a simple example of network in which an event realization leads to a non-simple path. We therefore adopt the following hypothesis: when a path crosses several times the same arc  $a$ , then we consider that delays encountered on the arc correspond to independent realizations of the random variable  $X_a$ .

Without loss of generality, we can assume that there exists in  $D$  at least one  $v$ - $d$  path (otherwise, the vertex  $v$  can simply be removed from  $D$ ). We have  $\lim_{t \rightarrow +\infty} F_v(t) = 1$  for all  $v \in V$  and  $F_v(\cdot)$  is non-decreasing. Thus,  $F_v(\cdot)$  can be considered as a cumulative distribution, and we can introduce a random variable  $U_v$  whose cumulative distribution function is  $F_v(\cdot)$ .

This construction shows that any algorithm solving Equation (1) solves the STOCHASTIC ON TIME ARRIVAL PROBLEM as well.

**Proposition 2.** *The  $U_v$ 's defined above satisfy Equation (1) and conversely, any solution of Equation (1) has cumulative distributions that satisfy Equation (2).*

*Proof.* The equivalence between (1) and (2) follows directly from the equality

$$F_{X \wedge Y}(t) = \max(F_X(t), F_Y(t)).$$

□

*Remark 1.* If all  $X_a$ 's are deterministic, the STOCHASTIC ON TIME ARRIVAL PROBLEM coincides with the usual deterministic shortest problem, for which by the way the solution computed a priori coincides with the solution computed during the trip.

**4.3. An algorithm.** We describe now a new polynomial algorithm computing a solution of Equation (1), and thus solving also the STOCHASTIC ON TIME ARRIVAL PROBLEM.

A random variable  $Z'_v$  and an integer  $t'_v$  are attached to each vertex  $v$  and updated during the algorithm. Initially,  $Z'_v = +\infty$  and  $t'_v = +\infty$  for each vertex  $v \neq d$ , while  $Z'_d = 0$  and  $t'_d = 0$ . During the algorithm, a queue  $L$  of vertices “to be expanded” is maintained. Initially, the queue  $L$  contains only  $d$ .

The algorithm ends when  $L$  is empty. While  $L$  is not empty, the following operations are repeated:

- Extract from  $L$  the vertex  $u$  with minimum  $t'_u$ . In case there are several such vertices, choose one with maximal value of  $F_{Z'_u}(t'_u)$ .
- Set  $t'_u = +\infty$ .
- For each arc  $(v, u)$  in  $\delta^-(u)$ , *expand*  $u$  along  $(v, u)$ : if  $Z'_v \not\leq_{st} X_{(v,u)} + Z'_u$ , then
  - Update  $t'_v = \min(t'_v, \min\{t : F_{Z'_v}(t) < F_{X_{(v,u)} + Z'_u}(t)\})$ .
  - Update  $Z'_v$  to  $Z'_v \wedge (X_{(v,u)} + Z'_u)$ .
  - Add  $v$  to  $L$  (if it is not already present).



**Proposition 3.** *This algorithm terminates in less than  $T|V|$  iterations, where  $T$  is the maximum length an elementary  $o$ - $d$  path can take (over all possible paths and over all possible events). Setting  $Z_v$  to be the last value of  $Z'_v$  for each  $v \in V$  provides then a solution of Equation (1).*

Algorithms in Nie & Fan (2006) and in Samaranayake *et al.* (2012) make a stronger assumption on arc distributions: they suppose the existence of a  $\delta > 0$  such that  $X_a \geq \delta$  for all arcs  $a$ . Moreover, the number of iterations of the algorithm in Nie & Fan (2006) is always  $T|V|$  and in Samaranayake *et al.* (2012) it is a  $\Omega(T|V|)$ , while in our approach, it corresponds to a worst case behavior: In general, we expect our algorithm to terminate after  $\gamma|V|$  iterations for a small  $\gamma$ . The worst  $\gamma$  encountered in the numerical experiments was 3.3, see Section 7 for more details.

*Proof of Proposition 3.* At any time during the algorithm, let  $Z''_v$  have the distribution of  $Z'_v$  the last time  $v$  was expanded and let  $t''_v$  be the value of  $t'_v$  right before being set to  $+\infty$  because of the expansion. If  $v$  has never been expanded, let  $Z''_v = +\infty$  and  $t''_v = -1$ . The update rule of  $t'_v$  ensures thus that

$$F_{Z''_v}(t) = F_{Z'_v}(t) \quad \text{for all } t \leq t'_v - 1.$$

Thus, if the algorithm terminates, we have  $t'_v = +\infty$  and  $Z''_v = Z'_v$  for each vertex  $v$ . As  $Z'_v$  is only updated when one of its outneighbor  $u$  is expanded, we have

$$(3) \quad Z'_v \stackrel{(d)}{=} \bigwedge_{(v,u) \in \delta^+(v)} (X_{(v,u)} + Z''_u).$$

Therefore  $Z'_v$  is then a solution to Equation (1).

It remains to prove that the algorithm terminates in a finite number of iterations.

We prove that  $\min_{v \in V} t'_v$  does not decrease along the algorithm, and that if this quantity remains constant for consecutive iterations, then  $\max_{v \in V} F_{Z'_v}(t'_v)$  does not increase.

Equation (3) implies that  $Z'_v \leq_{st} X_{(v,u)} + Z''_u$  for any  $(v,u) \in \delta^+(v)$ . Since  $F_{Z'_u}$  and  $F_{Z''_u}$  coincide up to  $t'_u - 1$ , we have

$$(4) \quad F_{Z'_v}(t) \geq F_{X_{(v,u)} + Z'_u}(t) \quad \text{for all } t \leq t'_u - 1.$$

Moreover, just before being updated because of the expansion of a vertex  $u$ , the integer  $t'_v$  is equal to  $t''_u$  because of the way the vertex  $u$  has been selected in  $L$ . This remark and Equation (3) implies together that after an expansion along an arc  $(v,u)$ , we have  $t'_v \geq t''_u$ .

Consider the algorithm right after a vertex  $u$  has been expanded.

Let  $v$  be such that  $(v,u) \in \delta^+(v)$ . If  $t'_v$  has just been updated to  $t''_u$ , then

$$F_{Z'_v}(t'_v) = \sum_{k=0}^{t''_u} \mathbb{P}(X_{(v,u)} = k) F_{Z'_u}(t''_u - k),$$

and since  $F_{Z'_u}$  is a nondecreasing map, we have  $F_{Z'_v}(t'_v) \leq F_{Z'_u}(t''_u)$ . If  $t'_v$  has not been updated while being already such that  $t'_v = t''_u$ , then  $F_{Z'_v}(t'_v) \leq F_{Z'_u}(t''_u)$  because of the selection rule of  $u$ . Therefore, the next vertex  $w$  to be expanded will either be such that  $t'_w > t''_u$ , or such that  $t'_w = t''_u$  and  $F_{Z'_w}(t'_w) \leq F_{Z'_u}(t''_u)$ .

To finish the proof of termination, we show now that  $t'_v \geq t''_v + 1$  at any time during the algorithm. If  $t'_v = +\infty$ , the inequality is clearly satisfied. Otherwise, let  $u$  be the last vertex that has been expanded and whose expansion has lead to an update of  $t'_v$ . We consider the algorithm right after this expansion.  $t''_u$  is thus the value of  $t'_u$  right before having being set to  $+\infty$ , and  $t'_v$  has been updated by the expansion.

Note that  $t''_u \geq t''_v$  since  $\min_{w \in V} t''_w$  does not decrease along the algorithm (it does not decrease because each  $t''_w$  is the value of  $\min_{v' \in V} t'_{v'}$  at some previous iteration, and this latter quantity is non decreasing, as already noted). We already know that  $t'_v \geq t''_u$ . Thus, if  $t'_v > t''_u$ , we have  $t'_v \geq t''_v + 1$  as required. It remains to check whether it is possible to have  $t'_v = t''_u = t''_v$  simultaneously. In such a case, we would necessarily have  $F_{Z'_v}(t''_v) \geq F_{Z'_u}(t''_u)$  since  $\max_{w \in V} F_{Z'_w}(t'_w)$  does not increase when  $\min_{w \in V} t'_w$  remains constant. Since  $F_{Z'_u}$  is a nondecreasing map, we would have  $F_{Z'_u}(t''_u) \geq F_{X_{(v,u)}+Z'_u}(t''_u)$  and thus

$$F_{Z'_v}(t) \geq F_{X_{(v,u)}+Z'_u}(t) \quad \text{for all } t \leq t''_u$$

with the help of Equation (4). Thus,  $t'_v$  should have been updated to a value greater than  $t''_u$ . Hence, in any case, we have  $t'_v \geq t''_v + 1$ .

Now, denote by  $T_v$  the maximum length an elementary  $v$ - $d$  path can take. We claim that once  $Z'_v$  has been updated for the first time, we have  $F_{Z'_v}(T_v) = 1$  all along the algorithm. It can be proved by a direct induction on the number of arcs an elementary path may have with the help of Equation (3). Moreover, since the inequality  $t'_v \geq t''_v + 1$  holds along the algorithm, we may have  $F_{Z'_v}(t) < F_{X_{(v,u)}+Z'_u}(t)$  only for  $t > t''_v$ . Therefore, after at most  $T_v$  iterations, the test “ $Z'_v \not\leq_{st} X_{(v,u)} + Z'_u$ ” is always false. It implies that after at most  $\max_{v \in V} T_v$  iterations, the algorithm terminates.  $\square$

*Remark 2.* For each elementary path  $P$ , define  $T_P$  to be the maximum length it can takes over all possible realizations. Denote by  $\mathcal{P}_{vd}$  the set of all elementary  $v$ - $d$  paths. Proposition 3 is actually true for  $T = \min_{P \in \mathcal{P}_{od}} T_P$ . Indeed, in the proof above, we show that the algorithms terminates after a finite number of iterations. Denote  $\bar{T}_v = \min_{P \in \mathcal{P}_{vd}} T_P$ . Proposition 1 ensures that  $F_{Z_v}(\bar{T}_v) = 1$ . Thus, as soon as  $t''_v = \bar{T}_v$ , the test “ $Z'_v \not\leq_{st} X_{(v,u)} + Z'_u$ ” is always false. It implies that after at most  $\max_{v \in V} \bar{T}_v$  iterations, the algorithm terminates.

*Remark 3.* When all  $X_a$  are deterministic, this algorithm coincides with Dijkstra’s algorithm.

## 5. STOCHASTIC $\rho$ -SHORTEST PATH PROBLEM

**5.1. Complexity.** When  $\rho$  is linear over independent random variables and dealt with as an oracle, the STOCHASTIC  $\rho$ -SHORTEST PATH PROBLEM can clearly be solved in polynomial time by any Dijkstra-like algorithm. However, we were not able to decide whether the problem remains polynomial in the general case, still dealing with  $\rho$  as an oracle. Nevertheless, we are able to prove the following theorem.

**Theorem 4.** *There is no polynomial algorithms with a complexity function independent of  $\rho$  solving the STOCHASTIC  $\rho$ -SHORTEST PATH PROBLEM, unless  $P = NP$ .*

*Proof.* The proof consists in describing a polynomial reduction of the deterministic resource constraint path problem to some STOCHASTIC  $\bar{\rho}$ -SHORTEST PATH PROBLEM, with an adequate risk measure  $\bar{\rho}$  consistent with the usual stochastic order. The deterministic resource constrained path problem consists in a directed graph  $D = (V, A)$ , two vertices  $o, d \in V$ , nonnegative integers  $(c_a)_{a \in A}$  (the costs), nonnegative integers  $(r_a)_{a \in A}$  (the resources), and a nonnegative integer  $R$  (the capacity). It aims at finding a path  $P$  with  $\sum_{a \in P} r_a \leq R$  and with minimal cost  $\sum_{a \in P} c_a$ . It is an NP-hard problem, see Handler & Zang (1980).

To that purpose, we define two risk measures  $\rho_{\min}$  and  $\rho_{\max}$ .

$$\rho_{\min}(X) = \min\{t \in \mathbb{Z}_+ : F_X(t) > 0\}$$

$$\rho_{\max}(X) = \begin{cases} 0 & \text{if } \max\{t \in \mathbb{Z}_+ : \mathbb{P}(X = t) > 0\} \leq M \\ 1 & \text{otherwise,} \end{cases}$$



where  $M = R(1 + \max_{a \in A} c_a)$ . We define

$$\bar{\rho} = \rho_{\min} + \left( \sum_{a \in A} c_a \right) \rho_{\max}.$$

The risk measure  $\rho_{\min}$  is consistent with the usual stochastic order because  $F_X(t) \geq F_Y(t)$  for all  $t$  implies  $\min\{t \in \mathbb{Z}_+ : F_X(t) > 0\} \leq \min\{t \in \mathbb{Z}_+ : F_Y(t) > 0\}$ . As well as  $\rho_{\min}$ , the risk measure  $\rho_{\max}$  is consistent with the usual stochastic order because  $\max\{t \in \mathbb{Z}_+ : \mathbb{P}(X = t) > 0\} = \min\{t \in \mathbb{Z}_+ : F_X(t) = 1\}$  and  $F_X(t) \geq F_Y(t)$  for all  $t$  implies  $\min\{t \in \mathbb{Z}_+ : F_X(t) = 1\} \leq \min\{t \in \mathbb{Z}_+ : F_Y(t) = 1\}$ . Thus  $\bar{\rho}$  is also consistent with the usual stochastic order.

We describe now the reduction to the STOCHASTIC  $\bar{\rho}$ -SHORTEST PATH PROBLEM. Given an arc  $a$ , we define  $X_a$  as follows:

$$\mathbb{P}(X_a = c_a) = \frac{1}{2} \quad \text{and} \quad \mathbb{P}\left(X_a = r_a(1 + \max_{b \in A} c_b)\right) = \frac{1}{2}.$$

$X_a$  can only take two values.

The deterministic resource constrained shortest path problem has a feasible solution if and only if the STOCHASTIC  $\bar{\rho}$ -SHORTEST PATH PROBLEM has a solution of cost less than  $\sum_{a \in A} c_a$ . In this case, the optimal solutions of both problems coincide, as we show now.

We assume that the deterministic resource constrained shortest path problem has a feasible solution. Let  $P$  be an optimal solution of the STOCHASTIC  $\bar{\rho}$ -SHORTEST PATH PROBLEM. We have then  $\bar{\rho}(\sum_{a \in P} X_a) = \sum_{a \in P} c_a$  and  $\sum_{a \in P} r_a \leq R$ , which implies that the deterministic resource constrained shortest path problem has a feasible solution of cost  $\sum_{a \in P} c_a$ . Conversely, an optimal solution  $P'$  of the deterministic resource constrained shortest path problem provides a feasible solution of the STOCHASTIC  $\bar{\rho}$ -SHORTEST PATH PROBLEM of cost  $\sum_{a \in P'} c_a$ .  $\square$

Note the  $\bar{\rho}$  used in the proof being fixed, the STOCHASTIC  $\rho$ -SHORTEST PATH PROBLEM becomes polynomially solvable. The proof above works precisely because the complexity function must be independent of the risk measure.

**5.2. Algorithm.** For any path  $P$ , we denote by  $X_P$  the random variable  $\sum_{a \in P} X_a$ . Sometimes in the proofs or in the computations, the path  $P$  can be non-simple. In this case, independent copies of the  $X_a$ 's appearing several times in the sum are used. We assume given for each vertex  $v$  a random variable  $Z_v^{LB}$  such that  $Z_v^{LB} \leq_{st} X_P$  for all  $v$ - $d$  paths  $P$ . Such a collection of random variables  $(Z_v^{LB})_{v \in V}$ , which play the role of stochastic lower bounds, can be computed for instance by the techniques presented in Section 4.

These stochastic lower bounds are used to discard some partial paths that are not subpaths of an optimal path. It resembles classical techniques used by shortest path algorithms in the deterministic setting, see for instance Bast *et al.* (2014) for state of the art techniques, in which the lower bound is a real number, while in our setting the lower bound is a random variable.

We present the algorithm, which is a labeling one. A *label*  $\lambda$  is a pair  $(v_\lambda, Y_\lambda)$ , where  $v_\lambda$  is a vertex and  $Y_\lambda$  is a random variable. Labels are stored in a queue  $L$ . Initially,  $L$  contains a unique label  $\lambda_o = (o, 0)$ . If there is no  $o$ - $d$  paths, the algorithm returns nothing. Otherwise, an elementary  $o$ - $d$  path  $P_0$  is computed and  $\rho_{od}^{UB}$  is set to  $\rho(X_{P_0})$ . The algorithm ends when  $L$  is empty. While  $L$  is not empty, the following operations are repeated:

- Extract a label  $\lambda$  of  $L$ .
- If  $v_\lambda = d$  and  $\rho(Y_\lambda) < \rho_{od}^{UB}$ : update  $\rho_{od}^{UB}$  to  $\rho(Y_\lambda)$ .
- Otherwise: If  $\rho(Y_\lambda + Z_{v_\lambda}^{LB}) < \rho_{od}^{UB}$ , *expand* label  $\lambda$ : for each arc  $(v_\lambda, v)$  in  $\delta^+(v_\lambda)$ , add a new label  $(v, Y)$  to  $L$ , where  $Y \stackrel{(d)}{=} Y_\lambda + X_{(v_\lambda, v)}$ .

**Theorem 5.** *Suppose that  $\mathbb{P}(X_a \neq 0) > 0$  for all  $a \in A$  and that there exists at least one solution. If  $\rho$  is consistent with the usual stochastic order, then the algorithm described above terminates after a finite number of iterations and at the end,  $\rho_{od}^{UB}$  is the optimal value of a solution of the STOCHASTIC  $\rho$ -SHORTEST PATH PROBLEM.*

Before giving the proof, we give the two ideas the algorithm relies on. Note that a label  $\lambda$  corresponds actually to some  $o$ - $v_\lambda$  path  $P_\lambda$  that the algorithm tries to expand in an optimal path (more details are given in the proof below).

The first idea is the following. At any time during the algorithm,  $\rho_{od}^{UB}$  is an upper bound on the optimal cost. Suppose that  $\rho_{od}^{UB}$  is not tight. If  $P$  is an  $o$ - $v$  subpath of an optimal solution, then

$$(5) \quad \rho(X_P + Z_v^{LB}) < \rho_{od}^{UB}.$$

In an enumeration of all the  $o$ - $d$  paths to find an optimal path, any partial path  $P$  that does not satisfy Equation (5) can thus be discarded.

The second idea is that, when an  $o$ - $d$  path  $P$  is such that  $\rho(X_P) < \rho_{od}^{UB}$ , then  $\rho_{od}^{UB}$  can be updated to  $\rho(X_P)$ . Such an update reduces the number of paths satisfying Equation (5).

As it is described, the algorithm computes only the optimal cost of a solution. The algorithm can easily be adapted to find the optimal path itself, simply by maintaining, for each label  $\lambda$  the label whose expansion has lead to its creation.

*Remark 4.* It is easy to deal with the presence of variables  $X_a$  with  $X_a = 0$ . If  $X_{(u,v)} = 0$ , remove  $(u, v)$  and add arcs  $(w, v)$  for  $w \in N^-(u)$  with  $X_{(w,v)} \stackrel{(d)}{=} X_{(w,u)}$  as well as arcs  $(u, w)$  for  $w \in N^+(v)$  with  $X_{(u,w)} \stackrel{(d)}{=} X_{(v,w)}$ .

*Proof of Theorem 5.* We associate to each label  $\lambda$  an  $o$ - $v_\lambda$  path  $P_\lambda$  such that  $Y_\lambda = X_{P_\lambda}$ : If  $P_\mu$  is associated to the label  $\mu$  and if  $\lambda$  is obtained by expanding  $\mu$ , we define  $P_\lambda$  as the path  $P_\mu + (v_\mu, v_\lambda)$ . It is easy to see that there is at most one label associated to a given path. This remark will be useful in the proof.

Initially,  $\rho_{od}^{UB}$  is equal to  $\rho(X_{P_0})$ , where  $P_0$  is the elementary  $o$ - $d$  path computed at the beginning of the algorithm. Besides,  $\rho_{od}^{UB}$  can only decrease during the algorithm. In addition, we have  $\rho(X_{P_\lambda}) \leq \rho(Y_\lambda + Z_{v_\lambda})$  by consistency of  $\rho$  with the usual stochastic order, and because  $X_{P_\lambda} \stackrel{(d)}{=} Y_\lambda \leq_{st} Y_\lambda + Z_{v_\lambda}^{LB}$ . Thus, as only labels such that  $\rho(Y_\lambda + Z_{v_\lambda}^{LB}) < \rho_{od}^{UB}$  are expanded, and as, given a path  $Q$ , there is at most one label such that  $P_\lambda = Q$ , the number of labels that are expanded during the algorithm is upper-bounded by the number of paths  $P$  such that  $\rho(X_P) < \rho(X_{P_0})$ .

To prove that the number of paths  $P$  satisfying  $\rho(X_P) < \rho(X_{P_0})$  is finite, we need the following claim.

*Let  $X$  and  $Y$  be random variables satisfying the assumption of Section 1.3 and denote by  $S_k$  be the sum of  $k$  independent copies of  $X$ . Suppose that  $\mathbb{P}(X \neq 0) > 0$ . Then there exists an integer  $n$  such that  $Y \leq_{st} S_n$ .*

This claim is proved as follows. Given an integer  $t$ , the event  $S_k \leq t$  requires that at least  $k - t$  copies of  $X$  are equal to 0. Using this idea and then the Stirling formula, we obtain  $F_{S_k}(t) \leq \binom{k}{t} P(X = 0)^{k-t} = O(k^t P(X = 0)^{k-t})$ . For any integer  $t$ , we have thus  $\lim_{k \rightarrow +\infty} F_{S_k}(t) = 0$ . It implies that for any  $t \leq t_{\max}^Y$ , where  $t_{\max}^Y$  is the maximum value  $Y$  can take, there is a  $k_t$  such that  $F_{S_k}(t) < F_Y(t)$  when  $k \geq k_t$ . The quantity  $t_{\max}^Y$  is finite, because of the finiteness of the support of  $Y$ . Defining  $n = \max_{0 \leq t \leq t_{\max}^Y} k_t$  proves the claim.

As a consequence of the claim, there is only a finite number of paths  $P$  such that  $X_{P_0} \not\leq_{st} X_P$ . As  $X_{P_0} \leq_{st} X_P$  implies  $\rho(X_{P_0}) \leq \rho(X_P)$  by consistency of  $\rho$  with the usual stochastic order, there is only a finite number of paths  $P$  such that  $\rho(X_P) < \rho(X_{P_0})$ . Hence, only a finite number of labels are expanded and the algorithm terminates after a finite number of iterations.

We prove now that the algorithm returns the correct value. Note that  $\rho_{od}^{UB}$  is at any time the cost of some  $o$ - $d$  path. Let  $P^*$  be the  $o$ - $d$  path that has updated  $\rho_{od}^{UB}$  for the last time. Note that  $P^*$  is not necessarily elementary.

Suppose that there is at least one  $o$ - $d$  path, otherwise the algorithm would return  $+\infty$ . Denote by  $P$  any elementary  $o$ - $d$  path. There are subpaths  $Q$  of  $P$  starting at  $o$  and for which there exists a label  $\lambda$  such that  $Q = P_\lambda$ . The path reduced to the vertex  $o$  is one of them. Let  $P_1$  be such a subpath with the largest number of arcs,  $\mu$  be such that  $P_1 = P_\mu$ , and  $v$  be the destination of  $P_1$ . We also denote by  $P_2$  the  $v$ - $d$  subpath of  $P$ . Since  $P_1$  has been chosen with a largest number of arcs, the label  $\mu$  has not been expanded. If  $v = d$ , we have  $\rho(X_P)$  larger than or equal to the last value of  $\rho_{od}^{UB}$ . Otherwise,  $v \neq d$  and we had  $\rho(Y_\mu + Z_v^{LB}) \geq \rho_{od}^{UB}$  when  $\mu$  has been extracted. Since  $\rho_{od}^{UB}$  is non-increasing during the algorithm and since we have  $X_{P_2} \geq_{st} Z_v^{LB}$ , the cost  $\rho(X_P)$  is necessarily larger than or equal to the last value of  $\rho_{od}^{UB}$ , which is larger or equal to the cost of some elementary  $o$ - $d$  path:  $P^*$  itself if it is elementary, or the elementary subpath obtained by removing the cycles otherwise. Thus the last value of  $\rho_{od}^{UB}$ , which is equal to  $\rho(P^*)$ , is the optimal cost.  $\square$

### 5.3. Speeding up the algorithm.

**5.3.1. Keys.** We can speed up the algorithm by making  $L$  a priority queue and using a key equal to  $\rho(Y_\lambda + Z_{v_\lambda}^{LB})$  for the element  $(v_\lambda, Y_\lambda)$ . The idea behind this choice is that we give priority to the label that seems to be the most promising, as  $\rho(Y_\lambda + Z_{v_\lambda}^{LB})$  is a lower bound on the cost of any  $o$ - $d$  having  $P_\lambda$  as subpath. The efficiency of the methods depends on the quality of the stochastic lower bound  $Z_{v_\lambda}^{LB}$ . In such a way, we may discard more partial paths, since the upper bound  $\rho_{od}^{UB}$  will likely be smaller than without the use of such keys.

**5.3.2. Upper bounds.** Suppose that, in addition to the lower bounds  $Z_v^{LB}$ , we maintain for each vertex  $v$  a list  $\ell_v = (X_{P_1}, X_{P_2}, \dots, X_{P_k})$ , where the  $P_i$  are elementary  $v$ - $d$  paths. Then, these paths can be used to update the minimum known cost during the algorithm. Let  $\lambda$  be a label. The path  $P_\lambda$  is an  $o$ - $v_\lambda$  path. As a consequence, for any distribution  $X_{P_i} \in \ell_{v_\lambda}$ , path  $P_\lambda + P_i$  is an  $o$ - $d$  path. Thus, if  $\rho_{od}^{UB} > \rho(X_\lambda + X_{P_i})$ , we have identified a path whose cost is smaller than the current upper bound on the cost of the optimal path, and  $\rho_{od}^{UB}$  can be updated to  $\rho(X_\lambda + X_{P_i})$ . Improving the upper bound during the algorithm enables to reduce the number of labels considered, and thus speeds up the algorithm.

The lists  $\ell_v$  can be built heuristically during some preprocessing. In our experiments, we built them during the computation of the lower bounds  $Z_v^{LB}$  with the STOCHASTIC ON TIME ARRIVAL PROBLEM algorithm.

## 6. STOCHASTIC $\rho$ -RESOURCE CONSTRAINED SHORTEST PATH PROBLEM

**6.1. Complexity.** The STOCHASTIC  $\rho$ -RESOURCE CONSTRAINED SHORTEST PATH PROBLEM is NP-hard, since the deterministic case is a special case and is already NP-hard.

**6.2. Algorithm.** As in Section 5.2, we denote by  $X_P$  the random variable  $\sum_{a \in P} X_a$ , where independent copies of  $X_a$ 's appearing several time in the sum are used, and we assume given for each vertex  $v$  a random variable  $Z_v^{LB}$  such that  $Z_v^{LB} \leq_{st} X_P$  for each  $v$ - $d$  path  $P$ . We assume moreover given for each vertex  $v$  the optimal cost  $\pi_v$  of an unconstrained  $v$ - $d$  path. The costs  $(\pi_v)_{v \in V}$  can

be efficiently computed using Dijkstra's algorithm.

The labeling algorithm for the STOCHASTIC  $\rho$ -RESOURCE CONSTRAINED SHORTEST PATH PROBLEM goes as follows. A label  $\lambda$  is a triple  $(v_\lambda, Y_\lambda, c_\lambda)$  where  $v_\lambda$  is a vertex,  $Y_\lambda$  is a random variable, and  $c_\lambda \in \mathbb{R}$  is a cost. Labels are stored in a queue  $L$ . Initially,  $L$  contains a unique label  $\lambda_o = (o, 0, 0)$ , and  $c_{od}^{UB} = 1 + \sum_{a \in A} c_a$ . The algorithm ends when  $L$  is empty. While  $L$  is not empty, the following operations are repeated:

- Extract a label  $\lambda$  of  $L$ .
- If  $v_\lambda = d$  and if the inequalities  $\rho(Y_\lambda) \leq \rho_0$  and  $c_\lambda < c_{od}^{UB}$  hold: update  $c_{od}^{UB}$  to  $c_\lambda$ .
- Otherwise: If  $\rho(Y_\lambda + Z_v^{LB}) \leq \rho_0$  and  $c_\lambda + \pi_{v_\lambda} < c_{od}^{UB}$ , *expand* label  $\lambda$ : for each arc  $(v_\lambda, v)$  in  $\delta^+(v_\lambda)$ , add a new label  $(v, Y_\lambda + X_{(v_\lambda, v)}, c_\lambda + c_{(v_\lambda, v)})$  to  $L$ .

In the following theorem, we use the notation “ $\sup \rho > \rho_0$ ”, which means that there exists a random variable  $Y$  such that  $\rho(Y) > \rho_0$ .

**Theorem 6.** *Suppose that for each arc  $a$  we have  $c_a \neq 0$  or  $\mathbb{P}(X_a \neq 0) > 0$  and that  $\sup \rho > \rho_0$ . If  $\rho$  is consistent with the usual stochastic order, then the algorithm described above terminates after a finite number of iterations and at the end, either  $c_{od}^{UB} = 1 + \sum_{a \in A} c_a$  and the STOCHASTIC  $\rho$ -RESOURCE CONSTRAINED SHORTEST PATH PROBLEM admits no feasible solutions, or  $c_{od}^{UB}$  is finite and equal to the optimal value of a solution of the STOCHASTIC  $\rho$ -RESOURCE CONSTRAINED SHORTEST PATH PROBLEM.*

In the case when  $\sup \rho \leq \rho_0$ , any path is feasible, and the optimal solution of the STOCHASTIC  $\rho$ -RESOURCE CONSTRAINED SHORTEST PATH PROBLEM is the unconstrained shortest path, which can be computed with Dijkstra's algorithm.

As noted in Section 5.2, this algorithm can easily be adapted to return a shortest path itself, and to deal with arcs  $a$  such that  $c_a = 0$  and  $X_a = 0$  by the same kind as the one presented in Remark 4.

We do not give the proof of Theorem 6 since it is very similar to the one of Theorem 5. We nevertheless explain three ideas on which the algorithm relies. They are very similar to the ones used in Section 5.2 for the STOCHASTIC  $\rho$ -RESOURCE CONSTRAINED SHORTEST PATH PROBLEM. Again, a label  $\lambda$  corresponds to some  $o$ - $v_\lambda$  path  $P_\lambda$  that the algorithm tries to expand in an optimal path

The first idea is the following. If an  $o$ - $v$  path  $P$  is a subpath of a feasible  $o$ - $d$  path, then

$$\rho(X_P + Z_v^{LB}) \leq \rho_0.$$

This inequality provides thus a condition that allows to discard partial path that cannot be completed into feasible paths.

The second idea is the following. Assume that  $c_{od}^{UB}$  is an upper bound on the optimal cost of the STOCHASTIC  $\rho$ -RESOURCE CONSTRAINED SHORTEST PATH PROBLEM. Suppose that  $c_{od}^{UB}$  is not tight. If  $P$  is an  $o$ - $v$  path subpath of an optimal solution, then

$$c_P + \pi_v < c_{od}^{UB}.$$

It can be used to discard partial paths that cannot be completed into optimal paths.

The third idea is the following. Each time a feasible  $o$ - $d$  path  $P$  satisfying  $c_P < c_{od}^{UB}$  is encountered, we have met a better feasible path and  $c_{od}^{UB}$  can be updated to  $c_P$ .

**6.3. Speeding up the algorithm.** The same techniques as the ones described in Section 5 can be used to speed up the algorithm.

**6.3.1. Keys.** The cost  $c_\lambda + \pi_{v_\lambda}$ , can be used as a key for the element  $v_\lambda$ . It represents a lower bound on the best cost that could be obtained with a path extending  $P_\lambda$ .

6.3.2. *Upper bounds on the complete path.* Compared to the STOCHASTIC  $\rho$ -SHORTEST PATH PROBLEM of Section 5, the only difference in the context of the STOCHASTIC  $\rho$ -RESOURCE CONSTRAINED SHORTEST PATH PROBLEM is that a list of pairs  $(X_{P_i}, c_{P_i})$  must be stored for each vertex  $v$  instead of a list of sole random variables  $X_{P_i}$ .

## 7. EXPERIMENTAL RESULTS

The algorithms have been coded in C++. The priority queue was implemented using the data-structures `map` and `multimap` of the C++ standard library. Experiments were carried out on a MacBook Pro with a 2,5 GHz Intel Core i5 processor and 4 Go RAM. Fast Convolution products were performed using the BSD licensed Fast Fourier Transform library `KissFFT` (Bogerd, 2013).

7.1. **Instances.** The algorithms were tested on square grid networks of various sizes and distribution types. The origin is the upper left corner of the grid and the destination is the lower right corner of the grid. A rough description of these instances is available on Table 1. The name of each instance indicates the width of the grid, and the distribution. Three types of distributions have been considered: randomly generated generic distributions, lognormal distributions with randomly generated parameters, and gamma distributions with different sizes.

Instances can be found on the following webpage:

[http://cermics.enpc.fr/~parmentat/shortest\\_path/](http://cermics.enpc.fr/~parmentat/shortest_path/)

The remaining of this subsection is devoted to additional information regarding the way the instances were generated.

7.1.1. *Distributions of the  $X_a$ 's.* We describe the way the distributions of arc travel times  $X_a$  are built. For all three distributions, the minimum value  $X_a$  can take is uniformly drawn at random between 0 and 50. We denote this minimum by  $t_0$ .

The generic distributions are then built as follows. The size of the support is then drawn uniformly at random between 1 and  $2t_0$ . The quantity  $\mathbb{P}(X_a = t_0 + t)$  is set to  $\frac{r_t}{\sum_t r_t}$ , where  $r_t$  is generated by selecting at random some prefixed intervals, in which we draw uniformly at random the value of  $r_t$ . The prefixed intervals are chosen in a way that enforces strong variations in the value of the variances among the arcs.

Regarding the lognormal (resp. gamma) distributions, a maximum value for the mean  $M$  is first selected. For the lognormal distributions,  $M$  is set to  $2t_0$ , except for instance `g100L1`, for which it is set to  $4t_0$ . For the gamma distributions, we set  $M = 10$ . Then, a mean  $\mu$  is uniformly generated between 1 and  $M$ , and a standard deviation  $\sigma^2$  is uniformly generated in  $[M - \mu, 2M - \mu]$ . The quantity  $\mathbb{P}(X_a = t_0 + t)$  is set to  $\frac{r_t}{\sum_t r_t}$ , where  $r_t$  is the density of a lognormal (resp. gamma) distributions of mean  $\mu$  and variance  $\sigma^2$  at time  $t$ . When  $r_t$  becomes smaller than some threshold  $\varepsilon$ , it is set to 0. It ensures a finite-size support.

Since we are expecting the complexity of the operations on distributions to be correlated to the size of paths distributions, we provide in Table 1 the size  $\ell$  of the STOCHASTIC ON TIME ARRIVAL PROBLEM solution distribution support at the origin.

7.1.2. *Risk measures.* We made the experiments with two risk measures:  $\mathbb{P}(\cdot \geq \tau)$  and  $CVaR_\beta$ .

For the first one, we set  $\tau$  to be equal to  $\min\{t : F_{Z_o}(t) \geq p\}$  for some parameter  $p$ , where  $Z_o$  is the solution of the STOCHASTIC ON TIME ARRIVAL PROBLEM at the origin. For the STOCHASTIC  $\rho$ -SHORTEST PATH PROBLEM,  $p$  was chosen in  $\{0.5, 0.8, 0.95\}$ . For the STOCHASTIC  $\rho$ -RESOURCE CONSTRAINED SHORTEST PATH PROBLEM,  $p$  was set to 0.95.

We did so because if  $\tau$  is too small, then  $\mathbb{P}(X_P \geq \tau) = 1$  for each path  $P$  from origin to destination, which implies that all the solutions of the STOCHASTIC  $\rho$ -SHORTEST PATH PROBLEM have the same solutions, and that the STOCHASTIC  $\rho$ -RESOURCE CONSTRAINED SHORTEST PATH



Instance	$ V $	$ A $	$\ell$	Dist. type
g10R	100	360	191	Generic
g40R	1600	6240	953	Generic
g100R	10000	39600	2449	Generic
g10Ls	100	360	138	Lognormal
g40Ls	1600	6240	389	Lognormal
g100Ls	10000	39600	797	Lognormal
g100Ll	10000	39600	2091	Lognormal - long
g10G	100	360	157	Gamma
g40G	1600	6240	538	Gamma
g100G	10000	39600	1301	Gamma

TABLE 1. Grid instances description.

PROBLEM is infeasible. On the contrary, when  $\tau$  is too large, all elementary origin-destination paths have the same risk measure 0, which implies that they are all optimal solutions of the STOCHASTIC  $\rho$ -SHORTEST PATH PROBLEM, and that the STOCHASTIC  $\rho$ -RESOURCE CONSTRAINED SHORTEST PATH PROBLEM is practically an unconstrained deterministic shortest path problem.

For the second risk measure, we made the experiments for the STOCHASTIC  $\rho$ -SHORTEST PATH PROBLEM with  $\beta \in \{0.01, 0.05, 0.25\}$ . The values 0.01 and 0.05 are usual values of the parameter. While the value 0.25 is higher than what is usually used for the *CVaR*, we think that in the context of the STOCHASTIC  $\rho$ -SHORTEST PATH PROBLEM, such a value corresponds to the expected travel time of the worst day in a week, which sounds reasonable when a commuter searches an optimal path. For the STOCHASTIC  $\rho$ -RESOURCE CONSTRAINED SHORTEST PATH PROBLEM,  $\beta$  was set to 0.05.

**7.1.3. Additional parameters in the case of the STOCHASTIC  $\rho$ -RESOURCE CONSTRAINED SHORTEST PATH PROBLEM.** The STOCHASTIC  $\rho$ -RESOURCE CONSTRAINED SHORTEST PATH PROBLEM requires two additional parameters: the resource constraint  $\rho_0$  and the arc costs  $c_a$ . We set  $\rho_0$  to be equal to  $\alpha\rho(Z_o) + (1 - \alpha)\rho(X_Q)$  for some  $\alpha \in [0, 1]$ . We use  $\rho(Z_o)$  and  $\rho(X_Q)$ , where  $Z_o$  is again the solution of the STOCHASTIC ON TIME ARRIVAL PROBLEM at the origin, and where  $Q$  is the optimal solution of the unconstrained deterministic shortest path problem. For each instance, we solved the STOCHASTIC  $\rho$ -RESOURCE CONSTRAINED SHORTEST PATH PROBLEM for  $\alpha \in \{0.02, 0.1, 0.5\}$ . We did so in order to have feasible but yet nontrivial solutions. The costs  $c_a$  were generated randomly, using a uniform law between 1 and twice the smallest  $t$  such that  $\mathbb{P}(X_a = t) > 0$ .

**7.2. Stochastic On Time Arrival Problem.** Table 2 describes the performance of the STOCHASTIC ON TIME ARRIVAL PROBLEM algorithm. “Upd.” is the number of updates operations, which can be compared to the number of arcs, and “Exp.” is the number of expansion of vertices, which should be compared to the number of vertices in the graph. First note that the algorithm is able to solve the STOCHASTIC ON TIME ARRIVAL PROBLEM on instances with 10’000 vertices in between 1 and 10 seconds. On all the instances we tested, the number of expansions was smaller than  $3.3|V|$ , and often close to  $|V|$ , which is the number of expansions needed to check that a solution is a feasible solution of the STOCHASTIC ON TIME ARRIVAL PROBLEM. Finally, the ratio of the number of expansions divided by the number of vertices in the instance increases slowly with the size of the instances, which allows to presume that the performance of the algorithm will remain high on larger instances. The limiting factor in our experiments was the memory available.

*Remark 5.* Note that STOCHASTIC ON TIME ARRIVAL PROBLEM algorithm CPU times to derive bounds to solve STOCHASTIC  $\rho$ -SHORTEST PATH PROBLEM and STOCHASTIC  $\rho$ -RESOURCE CONSTRAINED SHORTEST PATH PROBLEM given in Tables 3 to 6 are larger than the those of Table 2.



Instance	Upd.	Exp.	CPU time (s)
g10R	398	111	0.0037
g40R	14060	3598	0.3944
g100R	103397	26095	6.0776
g10Ls	403	113	0.0061
g40Ls	11074	2838	0.2712
g100Ls	80291	20271	3.1419
g100Ll	129797	32764	14.4251
g10G	430	121	0.0053
g40G	13696	3513	0.2853
g100G	107802	27214	4.1233

TABLE 2. STOCHASTIC ON TIME ARRIVAL PROBLEM algorithm performances.

Indeed, this is due to the computation of non-dominated path distributions in order to obtain upper bounds as explained in Section 5.3.2.

**7.3. Stochastic  $\rho$ -Shortest Path Problem.** Table 3 presents our numerical results when  $\rho(\cdot) = \mathbb{P}(\cdot \geq \tau)$  is used, and Table 4 when  $\rho(\cdot) = CVaR_\beta(\cdot)$ . In Table 3, the value of the parameter  $p$  used to choose  $\tau$  has been displayed. The next columns provide the parameters  $\tau$  or  $\beta$ , and the cost of the lower bound obtained by the solution of STOCHASTIC ON TIME ARRIVAL PROBLEM (SOTA) at the origin and the upper bound obtained by applying  $\rho(\cdot)$  to the optimal solution of the deterministic shortest path with expectation as cost. The performance of the STOCHASTIC ON TIME ARRIVAL PROBLEM algorithm used as preprocessing is then given. The next columns indicates the number of labels treated and expanded by the algorithm. Finally, the next columns provide the CPU time for the label algorithm, the cost of the optimal solution, and the total CPU time, which is roughly equal the sum of the CPU time of the STOCHASTIC ON TIME ARRIVAL PROBLEM algorithm and the CPU time of the labeling algorithm. The additional time is due to memory allocation.

We note that the algorithm is able to solve STOCHASTIC  $\rho$ -SHORTEST PATH PROBLEM for all instances in less that one minute. The most time-consuming phase is always the solution of STOCHASTIC ON TIME ARRIVAL PROBLEM done in preprocessing. The label algorithm runs on most instances in less than one second. The number of labels expanded by the STOCHASTIC  $\rho$ -SHORTEST PATH PROBLEM turns out to be small, which indicates that the quality of the lower bounds provided by the STOCHASTIC ON TIME ARRIVAL PROBLEM algorithm is good. No clear correlations appear between the parameter  $\tau$  or  $\beta$  and the performance of the algorithm. The cost of the solution of STOCHASTIC ON TIME ARRIVAL PROBLEM, the cost of the optimal solution of STOCHASTIC  $\rho$ -SHORTEST PATH PROBLEM, and the cost of the heuristic solution obtained by taking the path with minimal expectation are within a small range.

We remark that when parameter  $p$  is set to be equal to 0.5, the path with minimum expected length often also minimizes  $\mathbb{P}(X \geq \tau)$ . This sounds reasonable, because when  $p = 0.5$ , we expect a path with small  $\mathbb{P}(X \geq \tau)$  to have a small median and thus a small expectation.

**7.4. Stochastic  $\rho$ -Resource Constrained Shortest Path Problem.** Table 5 contains the numerical experiments for the STOCHASTIC  $\rho$ -RESOURCE CONSTRAINED SHORTEST PATH PROBLEM with  $\rho(\cdot) = \mathbb{P}(\cdot \geq \tau)$  as risk measure, and Table 6 contains the results with  $\rho(\cdot) = CVaR_\beta(\cdot)$ . Both tables contain the tradeoff  $\alpha$  between the risk measure of the solution of STOCHASTIC ON TIME ARRIVAL PROBLEM, the cost of the optimal unconstrained path and the resource constraint  $\rho_0$ . The next column indicates the time consumed by the STOCHASTIC ON TIME ARRIVAL PROBLEM

Instance	$p$	$\tau$	SOTA $\mathbb{P}(\cdot \geq \tau)$	ESP $\mathbb{P}(\cdot \geq \tau)$	SOTA CPU (s)	$\lambda$ treat.	$\lambda$ exp.	SSPP CPU (s)	Opt. sol. $\mathbb{P}(\cdot \geq \tau)$	Total CPU (s)
g40R	0.500	1773	0.495	0.498	0.788	304	79	0.081	0.498	0.878
g40R	0.800	1817	0.197	0.218	0.772	304	82	0.055	0.203	0.836
g40R	0.950	1855	0.049	0.074	0.784	304	83	0.059	0.051	0.852
g100R	0.500	4492	0.497	0.502	12.153	794	201	0.484	0.502	12.695
g100R	0.800	4565	0.197	0.203	12.110	1466	370	0.913	0.200	13.080
g100R	0.950	4630	0.049	0.057	12.729	794	214	0.498	0.050	13.287
g40Ls	0.500	1242	0.488	0.502	0.550	304	79	0.031	0.502	0.588
g40Ls	0.800	1268	0.199	0.217	0.544	452	117	0.048	0.217	0.598
g40Ls	0.950	1297	0.047	0.061	0.544	304	81	0.030	0.051	0.582
g100Ls	0.500	2959	0.493	0.496	6.203	794	201	0.132	0.496	6.378
g100Ls	0.800	2999	0.197	0.203	6.254	794	201	0.140	0.203	6.436
g100Ls	0.950	3040	0.049	0.053	6.322	1082	273	0.176	0.050	6.542
g100Ll	0.500	3971	0.498	0.503	30.104	1765	444	1.339	0.503	31.512
g100Ll	0.800	4070	0.200	0.204	30.186	1260	320	0.936	0.203	31.191
g100Ll	0.950	4170	0.050	0.053	30.156	1577	407	1.162	0.051	31.387
g40G	0.500	658	0.485	0.485	0.539	301	79	0.052	0.485	0.599
g40G	0.800	680	0.190	0.192	0.535	301	79	0.048	0.192	0.590
g40G	0.950	701	0.050	0.052	0.539	301	81	0.048	0.050	0.594
g100G	0.500	1708	0.492	0.498	8.702	2679	673	0.928	0.498	9.679
g100G	0.800	1746	0.195	0.204	8.601	8256	2079	2.906	0.202	11.555
g100G	0.950	1782	0.049	0.058	8.624	8436	2139	2.992	0.052	11.666

TABLE 3. Performance of STOCHASTIC  $\rho$ -SHORTEST PATH PROBLEM algorithm with  $\mathbb{P}(\cdot \geq \tau)$  as risk measure

Instance	$\beta$	SOTA $CVaR$	ESP $CVaR$	SOTA CPU (s)	$\lambda$ treat.	$\lambda$ exp.	SSPP CPU (s)	Opt. sol. $CVaR$	Total CPU (s)
g40R	0.250	1837.192	1845.262	0.800	260	83	0.052	1838.437	0.860
g40R	0.050	1873.505	1889.451	0.776	258	87	0.053	1874.776	0.837
g40R	0.010	1899.092	1922.075	0.775	253	94	0.055	1901.063	0.839
g100R	0.250	4599.412	4603.778	12.871	784	211	0.495	4600.425	13.424
g100R	0.050	4663.326	4672.600	12.481	1076	303	0.715	4664.403	13.254
g100R	0.010	4711.355	4724.452	12.156	1076	315	0.715	4712.107	12.929
g40Ls	0.250	1283.573	1287.441	0.545	282	81	0.030	1285.220	0.582
g40Ls	0.050	1312.482	1319.501	0.543	273	86	0.031	1313.445	0.581
g40Ls	0.010	1336.079	1345.904	0.543	259	93	0.037	1336.710	0.587
g100Ls	0.250	3021.004	3022.701	6.274	731	203	0.125	3021.967	6.441
g100Ls	0.050	3062.436	3065.510	6.323	934	276	0.179	3063.256	6.560
g100Ls	0.010	3095.327	3099.824	6.265	612	205	0.116	3095.587	6.425
g100Ll	0.250	4124.031	4126.803	30.856	1269	335	0.959	4125.520	31.885
g100Ll	0.050	4224.099	4228.733	30.691	755	211	0.565	4225.118	31.325
g100Ll	0.010	4302.636	4309.323	30.432	737	214	0.556	4303.215	31.057
g40G	0.250	691.308	691.725	0.554	272	80	0.055	691.523	0.619
g40G	0.050	713.333	714.331	0.551	252	82	0.045	713.517	0.604
g40G	0.010	730.689	732.304	0.555	224	85	0.044	730.875	0.607
g100G	0.250	1765.151	1768.065	8.640	8051	2083	2.904	1766.616	11.592
g100G	0.050	1801.330	1807.005	8.651	12803	3301	4.625	1803.171	13.326
g100G	0.010	1829.399	1837.500	8.605	19187	4952	6.975	1831.498	15.628

TABLE 4. Performance of STOCHASTIC  $\rho$ -SHORTEST PATH PROBLEM algorithm with  $CVaR$  as risk measure

(SOTA) algorithm used in preprocessing. The three next columns provide the number of labels treated and expanded by the STOCHASTIC  $\rho$ -RESOURCE CONSTRAINED SHORTEST PATH PROBLEM (SRCSP) labeling algorithm and the CPU time of the labeling algorithm. The labeling algorithm was stopped after five minutes. The column  $\rho(P)$  is the value of the risk measure evaluated on the solution found, LB is a lower bound on its cost (only displayed when the optimal solution was not

Inst.	$\tau$	$\alpha$	SP cost	$\rho_0$	SOTA CPU (s)	$\lambda$ treat.	$\lambda$ exp.	SRCSP CPU (s)	$\rho(P)$	LB	$c_P$	gap %	Total CPU (s)
g10R	498	0.02	134.106	0.069	0.008	67	19	0.003	0.050		320.702	0.0	0.012
g10R	498	0.10	134.106	0.145	0.008	79	22	0.003	0.070		219.858	0.0	0.012
g10R	498	0.50	134.106	0.525	0.007	318	82	0.013	0.258		202.102	0.0	0.021
g40R	1855	0.02	584.055	0.068	0.733	1827	462	0.319	0.062		1283.772	0.0	1.063
g40R	1855	0.10	584.055	0.144	0.724	75110	19177	13.486	0.123		1205.852	0.0	14.220
g40R	1855	0.50	584.055	0.525	0.726	1355571	429418	300.000	0.479	856.668	1248.012	45.7	300.735
g10Ls	418	0.02	251.844	0.069	0.012	111	30	0.005	0.059		421.254	0.0	0.018
g10Ls	418	0.10	251.844	0.145	0.012	360	94	0.016	0.146		400.475	0.0	0.029
g10Ls	418	0.50	251.844	0.525	0.013	1662	427	0.078	0.516		371.532	0.0	0.092
g40Ls	1297	0.02	833.451	0.066	0.533	1147	290	0.103	0.061		1541.857	0.0	0.646
g40Ls	1297	0.10	833.451	0.143	0.531	119204	30376	10.991	0.120		1490.817	0.0	11.533
g40Ls	1297	0.50	833.451	0.524	0.547	2823376	861653	300.000	0.480	1220.566	1433.313	17.4	300.556
g10G	185	0.02	213.270	0.062	0.009	117	31	0.005	0.058		297.191	0.0	0.014
g10G	185	0.10	213.270	0.116	0.010	175	46	0.007	0.076		269.665	0.0	0.018
g10G	185	0.50	213.270	0.386	0.008	149	45	0.007	0.243		231.777	0.0	0.016
g40G	701	0.02	892.028	0.069	0.535	10528	2635	1.758	0.064		1577.910	0.0	2.305
g40G	701	0.10	892.028	0.145	0.527	1231773	307952	207.613	0.143		1473.075	0.0	208.151
g40G	701	0.50	892.028	0.525	0.553	1468578	551686	300.000	-	999.764	$\infty$	$\infty$	300.560

TABLE 5. Performance of STOCHASTIC  $\rho$ -RESOURCE CONSTRAINED SHORTEST PATH PROBLEM algorithm with  $\mathbb{P}(\cdot \geq \tau)$  as risk measure, with  $p = 0.95$

found), and  $c_P$  is its cost. When a cost equal to  $\infty$  is displayed, it means that the algorithm was not able to find any feasible solution. One of the instances was moreover proved to be not feasible (g10G of Table 6). Finally, we provide the total CPU time, which is roughly equal to the sum of the preprocessing CPU time and the labeling algorithm CPU time.

The performance of the algorithm is strongly influenced by the parameter  $\alpha$ , which models the hardness of the resource constraint. A small  $\alpha$  corresponds to a hard resource constraint and a large  $\alpha$  corresponds to an easy resource constraint. The algorithm performance relies on the use of stochastic lower bound to cut potentially infeasible path. Thus, when the resource constraint is hard, which corresponds to a small  $\alpha$ , the number of feasible paths is small, and the algorithm is efficient. The labeling algorithm was able to solve all the instances with  $\mathbb{P}(\cdot \geq \tau)$  and most instances with  $\rho(\cdot) = CVaR_\beta(\cdot)$  when  $\alpha = 0.02$  in the allocated time. On the contrary, when  $\alpha$  is larger, the number of feasible paths is much bigger, making the algorithm less efficient.

## REFERENCES

- Bast, Hannah, Delling, Daniel, Goldberg, Andrew, Müller-Hannemann, Matthias, Pajor, Thomas, Sanders, Peter, Wagner, Dorothea, & Werneck, Renato. 2014. *Route Planning in Transportation Networks*.
- Bogerdig, Mark. 2013. *KissFFT library*.
- Boland, Natashaia, Dethridge, John, & Dumitrescu, Irina. 2006. Accelerated label setting algorithms for the elementary resource constrained shortest path problem. *Operations Research Letters*, **34**(1), 58–68.
- Chen, Anthony, & Ji, Zhaowang. 2005. Path finding under uncertainty. *Journal of advanced transportation*, **39**(1), 19–37.
- Chen, Bi Yu, Lam, William HK, Sumalee, Agachai, Li, Qingquan, Shao, Hu, & Fang, Zhixiang. 2013. Finding reliable shortest paths in road networks under uncertainty. *Networks and spatial economics*, **13**(2), 123–148.
- Dumitrescu, Irina, & Boland, Natashaia. 2003. Improved preprocessing, labeling and scaling algorithms for the Weight-Constrained Shortest Path Problem. *Networks*, **42**(3), 135–153.

Inst.	$\alpha$	SP cost	$\rho_0$	SOTA CPU (s)	$\lambda$ treat.	$\lambda$ exp.	SRCSP CPU (s)	$\rho(P)$	LB	$c_P$	gap %	Total CPU (s)
g10R	0.02	134.106	509.474	0.007	67	19	0.002	505.811		219.858	0.0	0.010
g10R	0.10	134.106	533.220	0.007	382	98	0.016	513.358		202.102	0.0	0.025
g10R	0.50	134.106	651.949	0.008	612	244	0.035	645.537		150.467	0.0	0.044
g40R	0.02	584.055	1896.338	0.765	28474	7235	4.913	1889.475		1205.852	0.0	5.691
g40R	0.10	584.055	1987.669	0.727	1256561	427909	300.000	1987.238	793.711	1050.238	32.3	300.736
g40R	0.50	584.055	2444.322	0.731	670441	514178	300.000	2439.891	613.004	742.644	21.1	300.741
g10Ls	0.02	251.844	432.411	0.012	111	30	0.005	429.405		421.254	0.0	0.018
g10Ls	0.10	251.844	453.657	0.013	1050	268	0.049	445.882		374.680	0.0	0.064
g10Ls	0.50	251.844	559.885	0.012	1357	353	0.076	513.708		283.292	0.0	0.089
g40Ls	0.02	833.451	1334.621	0.521	94912	24051	8.796	1333.675		1490.817	0.0	9.328
g40Ls	0.10	833.451	1423.180	0.558	2520108	1025174	300.000	1417.246	1028.961	1336.761	29.9	300.569
g40Ls	0.50	833.451	1865.975	0.538	1202558	836889	300.001	1860.527	880.748	1007.203	14.4	300.548
g10G	0.02	213.270	192.069	0.009	101	26	0.004			INFEAS.		0.014
g10G	0.10	213.270	195.236	0.009	118	31	0.005	194.351		269.665	0.0	0.015
g10G	0.50	213.270	211.069	0.009	173	50	0.008	203.028		231.777	0.0	0.018
g40G	0.02	892.028	719.158	0.547	18664	4669	3.133	718.039		1562.206	0.0	3.690
g40G	0.10	892.028	742.458	0.562	1631484	505573	300.000	-	1121.193	$\infty$	$\infty$	300.573
g40G	0.50	892.028	858.958	0.573	839101	632986	300.000	857.531	923.304	1163.971	26.1	300.583

TABLE 6. Performance of STOCHASTIC  $\rho$ -RESOURCE CONSTRAINED SHORTEST PATH PROBLEM algorithm with  $CVaR_\beta$  with  $\beta = 0.05$  as risk measure

- Ehrgott, Matthias. 2005. *Multicriteria optimization*. Vol. 2. Springer.
- Eiger, Amir, Mirchandani, Pitu B, & Soroush, Hossein. 1985. Path preferences and optimal paths in probabilistic networks. *Transportation Science*, **19**(1), 75–84.
- Fan, Yueyue, & Nie, Yu. 2006. Optimal routing for maximizing the travel time reliability. *Networks and Spatial Economics*, **6**(3-4), 333–344.
- Fan, YY, Kalaba, RE, & Moore II, JE. 2005. Arriving on time. *Journal of Optimization Theory and Applications*, **127**(3), 497–513.
- Frank, H. 1969. Shortest paths in probabilistic graphs. *Operations Research*, **17**(4), 583–599.
- Fu, Liping. 2001. An adaptive routing algorithm for in-vehicle route guidance systems with real-time information. *Transportation Research Part B: Methodological*, **35**(8), 749–765.
- Fu, Liping, & Rilett, Larry R. 1998. Expected shortest paths in dynamic and stochastic traffic networks. *Transportation Research Part B: Methodological*, **32**(7), 499–516.
- Hall, Randolph W. 1986. The fastest path through a network with random time-dependent travel times. *Transportation science*, **20**(3), 182–188.
- Handler, Gabriel Y, & Zang, Israel. 1980. A dual algorithm for the constrained shortest path problem. *Networks*, **10**(4), 293–309.
- Irnich, Stefan, & Desaulniers, Guy. 2005. *Shortest path problems with resource constraints*. Springer.
- Kosuch, Stefanie, & Lissner, Abdel. 2010. Stochastic shortest path problem with delay excess penalty. *Electronic Notes in Discrete Mathematics*, **36**, 511–518.
- Loui, Ronald Prescott. 1983. Optimal paths in graphs with stochastic or multidimensional weights. *Communications of the ACM*, **26**(9), 670–676.
- Miller-Hooks, Elise, & Mahmassani, Hani. 2003. Path comparisons for a priori and time-adaptive decisions in stochastic, time-varying networks. *European Journal of Operational Research*, **146**(1), 67–82.
- Miller-Hooks, Elise D, & Mahmassani, Hani S. 1998. Least possible time paths in stochastic, time-varying networks. *Computers & operations research*, **25**(12), 1107–1125.
- Miller-Hooks, Elise Deborah. 1997. *Optimal routing in time-varying, stochastic networks: algorithms and implementations*. The University of Texas at Austin.

- Mirchandani, Pitu B. 1976. Shortest distance and reliability of probabilistic networks. *Computers & Operations Research*, **3**(4), 347–355.
- Müller, Alfred, & Stoyan, Dietrich. 2002. *Comparison methods for stochastic models and risks*. Vol. 389. Wiley.
- Murthy, Ishwar, & Sarkar, Sumit. 1996. A relaxation-based pruning technique for a class of stochastic shortest path problems. *Transportation Science*, **30**(3), 220–236.
- Murthy, Ishwar, & Sarkar, Sumit. 1998. Stochastic shortest path problems with piecewise-linear concave utility functions. *Management Science*, **44**(11-part-2), S125–S136.
- Nie, Yu, & Fan, Yueyue. 2006. Arriving-on-time problem: discrete algorithm that ensures convergence. *Transportation Research Record: Journal of the Transportation Research Board*, **1964**(1), 193–200.
- Nie, Yu Marco, & Wu, Xing. 2009. Shortest path problem considering on-time arrival probability. *Transportation Research Part B: Methodological*, **43**(6), 597–613.
- Nie, Yu Marco, Wu, Xing, & Homem-de Mello, Tito. 2012. Optimal path problems with second-order stochastic dominance constraints. *Networks and Spatial Economics*, **12**(4), 561–587.
- Niknami, Mehrdad, Samaranayake, Samitha, & Bayen, Alexandre. 2014. Tractable Pathfinding for the Stochastic On-Time Arrival Problem. *arXiv preprint arXiv:1408.4490*.
- Nikolova, Evdokia. 2010. High-performance heuristics for optimization in stochastic traffic engineering problems. *Pages 352–360 of: Large-Scale Scientific Computing*. Springer.
- Nikolova, Evdokia, Kelner, Jonathan A, Brand, Matthew, & Mitzenmacher, Michael. 2006. Stochastic shortest paths via quasi-convex maximization. *Pages 552–563 of: Algorithms–ESA 2006*. Springer.
- Raith, Andrea, & Ehrgott, Matthias. 2009. A comparison of solution strategies for biobjective shortest path problems. *Computers & Operations Research*, **36**(4), 1299–1331.
- Sabran, Guillaume, Samaranayake, Samitha, & Bayen, Alexandre M. 2014. Precomputation techniques for the stochastic on-time arrival problem. *Pages 138–146 of: ALLENEX*. SIAM.
- Samaranayake, Samitha, Blandin, Sebastien, & Bayen, A. 2012. A tractable class of algorithms for reliable routing in stochastic networks. *Transportation Research Part C: Emerging Technologies*, **20**(1), 199–217.
- Sivakumar, Raj A, & Batta, Rajan. 1994. The variance-constrained shortest path problem. *Transportation Science*, **28**(4), 309–316.
- Tarapata, Zbigniew. 2007. Selected multicriteria shortest path problems: An analysis of complexity, models and adaptation of standard algorithms. *International Journal of Applied Mathematics and Computer Science*, **17**(2), 269–287.

A. PARMENTIER, UNIVERSITÉ PARIS EST, CERMICS, 6-8 AVENUE BLAISE PASCAL, CITÉ DESCARTES, 77455 MARNE-LA-VALLÉE, CEDEX 2, FRANCE  
*E-mail address:* axel.parmentier@cermics.enpc.fr

F. MEUNIER, UNIVERSITÉ PARIS EST, CERMICS, 6-8 AVENUE BLAISE PASCAL, CITÉ DESCARTES, 77455 MARNE-LA-VALLÉE, CEDEX 2, FRANCE  
*E-mail address:* frederic.meunier@enpc.fr